DIPLOMARBEIT

Titel der Diplomarbeit

A Large Neighbourhood Search for the 2-Echelon Capacitated Vehicle Routing Problem

Verfasser

Ulrich Breunig

angestrebter akademischer Grad

Magister der Sozial- und Wirtschaftswissenschaften

(Mag. rer. soc. oec)

Wien, 2012

Studienkennzahl lt. Studienblatt:	A 157
Studienrichtung lt. Studienblatt:	Internationale Betriebswirtschaft
Betreuer:	O.Univ.Prof. Dipl-Ing. Dr. Richard F. Hartl

ACKNOWLEDGEMENTS

First of all I would like to express my sincere appreciation to the staff of the Chair of Production and Operations Management at the University of Vienna for their great support. I would especially like to thank Mag. Dipl.-Ing. Dr. Verena Schmid for her support - as well as her great knowledge and creative ideas for new solution approaches.

My deep gratitude goes to my parents and grandparents for their year-long support and financing of my studies, and I would like to especially thank my mother Mag. Eva Breunig for coaching me in any mathematics-related questions throughout my entire studies as well as for marrying my father Dipl.-Ing. Franz Breunig, whose brilliant programming skills helped me to solve more than just one problem.

I would also like to thank my girlfriend Lea for her patience and continuous support through all the moods I lived during stressful times.

My gratitude also goes to all my friends and in particular my flatmates for learning together and sharing a good college life, as well as to my friend and business partner Nikolaus for his understanding.

Vienna, March 2012

Contents

Abstract

City Logistics involves different concepts of delivering goods to customers within densely populated areas. Many of them involve a two-tiered setup, which then is modelled in terms of a Two-Echelon Capacitated Vehicle Routing Problem. Large trucks deliver goods from a depot to intermediate facilities, so called satellites, where freight is transferred to smaller vehicles (city freighters), which then deliver it to customers. The goal is to satisfy all customer demands with the lowest possible costs and driven distance. A local-search metaheuristic based on a Large Neighbourhood Search is developed and implemented to find good solutions within limited computing time. A mixed integer programming model is used to create benchmarks for evaluating the quality of the solutions found by the metaheuristic, at least for small instances, where computing time is sufficient to solve to optimality.

1 Introduction

Transportation of goods is an important factor for economies. Many newly industrialising countries are heavily investing in transport infrastructure. They need to set up a reliable link between producers, wholesalers, sales and customers. Industrialised countries, on the other hand, face the problem of transport arising as a growing nuisance, especially in cities. According to [Crainic, 2008, p. 1] "There are few activities going on in a city that do not require at least some commodities being shipped." The Organisation for Economic Co-operation and Development stated three new developments in freight transport:

> "Globalisation of economic activities, changes in consumer behaviour and developments in advanced technologies have led to many developments:

- Businesses have expanded the area of their sourcing and distribution operations, developing world-wide supply chains that link customers, suppliers and manufacturers. Urban goods transport has therefore become integrated with long-haul transport. Businesses seek to improve the flow of their supply chains by utilising information and communications technologies (ICT) and optimise such supply chains by reducing the number of warehouses, centralising inventory and consolidating deliveries.
- The retail sector seeks to minimise cost by saving storage space and reducing stock, resulting in strict demands being placed on the supply chain which include reduced delivery lead times and just-in-time deliveries.
- As customers become increasingly integrated in the supply chain, the need to respond more rapidly to varied and often-changing customer demand requires the flow of the supply chain to be increasingly time sensitive. The rapid development of e-commerce also requires fast and reliable delivery.

These developments have led to increases in freight transport and further increases are unavoidable if no additional measures are taken."[1]

Therefore a main goal of City Logistics has to be boosting the efficiency of transporting goods from suppliers to customers. This involves different aspects with different planning horizons. On an operational level this includes intelligent routing of vehicles and e.g. reducing the distance travelled, optimising fleet size, vehicle dimensions and characteristics. The wise selection of locations for production sites, warehouses and freight terminals is a typical strategic decision.

[1] [Organisation for Economic Co-operation and Development (OECD), 2003, p. 8]

Higher efficiency of transport has positive effects for companies, as transportation costs will be lower. Driving less kilometres helps reduce costs for fuel and other vehicle related costs as well as spending less on wages for drivers. Less consumed fuel has positive effects on the environment as the emissions of several greenhouse-gases are saved. Overall traffic on streets is reduced, so for large scale transportation optimisation one might avoid one or the other traffic congestion.

We will put emphasis on a two-tiered setup of City Logistics. Goods originate at a depot that can provide sufficient quantities to satisfy all customer demands. All goods pass through one of several satellites where they are transferred from a truck to a city freighter, from the first transport level to the second one. So the first level includes large trucks which deliver the goods from the depot to the satellites. City freighters are part of the second level. They receive their load from the trucks at the satellites and then deliver to customers. Both levels have to be synchronised with each other. As satellites do not provide any warehousing functionality, the incoming load has to correspond exactly the outgoing quantity by the city freighters. There are no split deliveries allowed, and no consolidation of freight between the satellites.

Calculating optimal or at least near optimal routes for the vehicles involved in the delivery of goods is obviously more complex than with direct routes from origin to customer, like in well studied setups like the Capacitated Vehicle Routing Problem (CVRP) alone. The underlying complexity of the Two-Echelon Capacitated Vehicle Routing Problem (2E-CVRP) is increased by adding an additional layer combined with the necessary constraints for the synchronisation of both levels. [Feliu et al., 2008, p. 15] stated that:

> "The problem is easily seen to be NP-Hard via reduction to CVRP, which is a special case of 2E-CVRP arising when just one satellite is considered."

This thesis is organised as follows: we present a short history of routing problems leading to the 2E-CVRP and give a more detailed problem description in Section 2. We provide an overview of recent publications on two-tiered City Logistic setups in Section 3. A mathematical model formulation is shown in Section 4. Section 5 describes the development of the proposed metaheuristic. In Section 6 we show the results obtained by the metaheuristic and compare them to solutions from previous publications. Section 7 concludes this study.

2 Problem Description

2.1 History of Routing Problems

One of the best known and studied problems in Operations Research is the Travelling Salesman Problem (TSP). It seems to have been first formulated by Menger [1932]. The setup is quite simple (even if the solution is not): Find the shortest tour visiting each city on a given list exactly once and return to the point of origin, where all pairwise distances between cities have to be known. A possible but not necessary optimal tour is shown in Figure 1.

In the 1940s some publications addressed the search for the optimal route to visit all 49 USA state capitals, pushing the mathematical formulation of the problem but failing to find a solution. The optimal tour starting from Washington DC was finally found by Dantzig et al. [1954], where they also stated a way of avoiding subcycles. The studies on the TSP go hand in hand with the possibility of using computers for fast execution of calculations, solving larger and larger problem sets. Crowder and Padberg [1980] solved an instance with 318 cities. Some years later, as programming techniques advanced, Padberg and Rinaldi [1987] were able to find an optimal solution for 532 cities. Applegate et al. [2001] solved a TSP linking all the 15.112 largest cities in Germany with the optimal

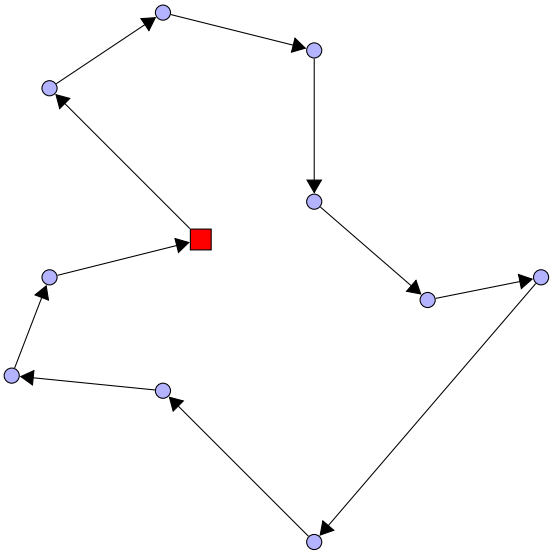

Figure 1: Travelling Salesman Problem

shortest route. The sizes of instances with known optimal solutions grew nearly exponentially, taking it to 85.900 points by Applegate et al. [2006].

Dantzig and Ramser [1959] extended the idea of the TSP. Their so-called "Truck Dispatching Problem" is a generalisation of the TSP in that it introduces additional conditions. The travelling salesman might have to return to his point of origin several times in between his route due to capacity constraints, which is shown in Figure 2.

As far as the problem is seen independent of time, the several generated routes could also be served at the same time by more than one salespersons - or vehicles. If clustering is done in the first step and each customer is assigned to a depot, the single routes are regarded as several independent TSPs. If the problem is seen, for example, as several vehicles delivering goods to customers, it does not matter which vehicle delivers to which customer, as long as all demands are satisfied. Another useful restriction may be the vehicle capacity. The sum of demands from all customers on one route must not exceed the maximum vehicle capacity.

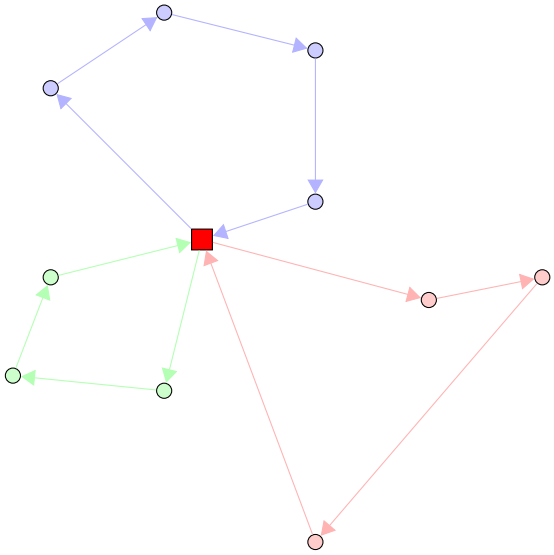

Figure 2: Vehicle Routing Problem

The next extension on our way to the 2E-CVRP is the Multi-Depot Vehicle Routing Problem (MD-VRP). There are more depots where vehicles are based. This is the only difference to the CVRP which has only one depot. A graphical representation of the solution of a MD-VRP can be seen in Figure 3.

2.2 Two-Echelon Capacitated Vehicle Routing Problem

If we think of the red depots in Figure 3 as warehouses and not as production facilities, it is obvious that the goods have to be delivered to these depots as well. So taking a look at solving this problem altogether it leads directly to the desired two-tiered setup of the 2E-CVRP. For this problem setup we will refer to the red squares as "satellites" which are supplied from one major depot (or production site), shown in Figure 4 as a black triangle.

The first level of transport is shown in black, representing trucks delivering goods from the depot to the satellites. This level can be modelled as a CVRP for delivering goods from one depot to the satellites. The second level of transport,

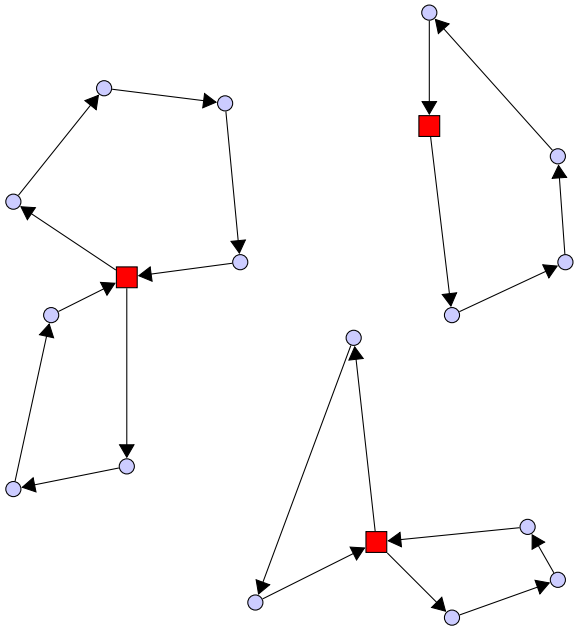

Figure 3: Multi-Depot Vehicle Routing Problem

shown in blue, represents the city freighters delivering goods from one of the satellites to the customers. Disregarding the fact that goods have to be delivered to satellites beforehand, this level can be modelled as a MD-VRP. So a crucial point of the 2E-CVRP is the synchronisation between these two levels. They are linked together at the satellites, where larger trucks have to unload exactly the quantity of goods that the city freighters then deliver to the customers. Changes in customer-to-satellite assignment on level two affect the quantities which have to be delivered to satellites by trucks. Both levels will have to be solved iteratively, as changes in one level always affect the other level as well. There are no split deliveries allowed and no consolidation of goods directly between satellites.

A two-tiered setup like this is of practical use in ancient European cities for example like Rome which has very narrow streets in the historic centre.[2] As goods are delivered in large trucks, they simply have to be reloaded to smaller

[2]cp. Crainic et al. [2004, 2009]

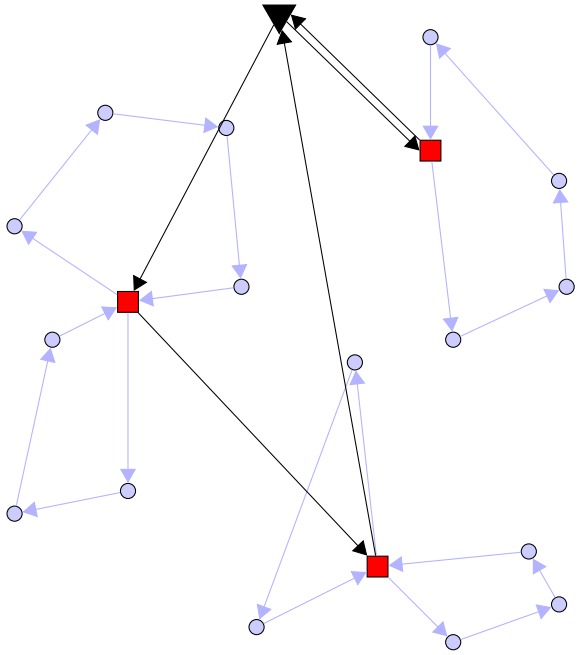

Figure 4: Two-Echelon Capacitated Vehicle Routing Problem

vehicles which are able to pass through the streets leading to the customers in the centre.

The first-level trucks could easily be exchanged by any other type of transport, such as train, ship or tramway. The main aspect of the two different levels used to deliver goods to customers remains unchanged. Amsterdam for example had a working two-tiered setup with cargo trams involved. The "citycargo" project was launched in 2007 and it was planned to use 50 cargo trams to deliver goods to the city centre and reduce truck traffic there.[3] Unfortunately, the operating company went bankrupt in 2009 after investors pulled out their money, and the company's website http://www.citycargo.nl is no longer available.

[3]http://nlarchitects.wordpress.com/2009/07/14/city-cargo/, 1.2.2012

3 Literature Overview

There exist only few publications for 2-echelon routing setups, compared to the well-studied and longer known problems such as the TSP or the CVRP. There were some publications dealing with the special case of Rome, where large trucks cannot enter the historic centre and smaller vehicles have to be used: [Crainic et al., 2004, p. 124] suggested the use of mini satellite platforms for good consolidation to small city freighters, without warehousing and no need for significant physical installations, Crainic et al. [2007] extended the work with the introduction of time-windows for the customers and proposed a general time-dependent formulation, but there was no implementation reported. Also Gragnani et al. [2004] dealt with the delivery of goods with the case of Rome. Feliu et al. [2008] introduced test instances for the 2E-CVRP generated from the existing instances for the CVRP by Christofides and Eilon [1969]. Crainic et al. [2008] implemented a clustering-based metaheuristic. They first assigned customers to satellites and then solved the remaining MD-VRP, but provided solutions only for very small test instances. Baldacci et al. [2011] proposed an exact method for the 2E-CVRP, but we have not received any final results of their work yet and the paper is still under revision.

Recently Hemmelmayr et al. [2011] published an Adaptive Large Neighbourhood Search (ALNS) with promising results, as it is fast and also solves large instances. Their setting allows split deliveries for the satellites and therefore cannot be compared to our results. Perboli et al. [2008] gave an overview of multi-echelon transportation systems and showed additional valid inequalities for exact methods of solving the 2E-CVRP.

The setting of the proposed metaheuristic is very similar to the work of Mancini et al. [2011], which has not been published yet. We extended their work by introducing setting fixed costs for the vehicles and different first-level truck-

specific transportation costs. Every truck or city freighter can cause additional fixed costs if it leaves its depot, or satellite. Transportation costs can be set independently for both levels, as larger trucks usually tend to result in higher transportation costs than smaller city freighters.

4 Mathematical Model

We will show a mathematical model for the 2E-CVRP without split deliveries. The basic model presupposes known deterministic demands of customers. Travel distances between all nodes are known. There is one homogeneous good, one type of truck and one type of city freighter, both with a known maximum capacity. There are three different types of locations: one *depot*, with unlimited capacity. As all customers demands have to be satisfied, the depot simply provides enough goods. Several *satellites*, where goods are transferred from trucks to city freighters without storage in between. This is the place where the first and second level are linked together and need to be synchronised. Not all of the satellites have to be used. The third type of locations are the *customers* with known demands.

The main design parameters include the following: The problem is static, all demands are deterministic. There is only one single product, originating at only one depot. There exist two different types of vehicles, both capacitated. The first echelon vehicles are called trucks and have a larger capacity than the second echelon vehicles, called city freighters. No split deliveries are allowed, so any vehicle has to provide the total amount of goods demanded by the served destination. There is also no direct consolidation of goods between the satellites. This leads to a maximum capacity of the satellites at the size of one truckload. All vehicles return to their home satellite or depot at the end of their route. The objective is minimisation of total costs.

4.1 Notation

All the nodes (depot, satellites and customers) are connected via a network of arcs with each other, where all travelling distances are known. We used a very common notation for routing problems, i.e. decision variables modelling the flow of vehicles on the arcs of the underlying network. The depot will be denoted as $\mathcal{D}$, the set of potential locations for the satellites as $\mathcal{S}$, and the set of customers' locations will be denoted as $\mathcal{C}$.

The first level of transport will be trucks leaving from depot to satellites, so the set $\mathcal{D} \cup \mathcal{S}$ is denoted as $\mathcal{A}^T$, the set for second level transport, i.e. vehicles from satellites to customers, $\mathcal{S} \cup \mathcal{C} = \mathcal{A}^V$. Each customer $i \in \mathcal{C}$ has a demand D_i.

The set of trucks will be denoted as $\mathcal{T}$, set of city freighters as $\mathcal{V}$.

C_{ij} is the travelling distance from node i to $j \quad \forall i, j \in \mathcal{D} \cup \mathcal{S} \cup \mathcal{C}$, which is directly proportional to transportation costs. $\alpha \geq 1$ is used for weighting the cost per distance travelled by a large sized and therefore more expensive truck, compared to the cost per distance travelled by a smaller city freighter.

C_T are fixed costs for using one truck, C_V fixed cost per used city freighter. As we consider two homogeneous fleets of vehicles, Q_T is the maximum capacity of a truck, Q_V is the maximum capacity of a city freighter.

Binary flow variables x^T_{ijt} evaluate to 1 if and only if truck t drives from node i to node j, and 0 otherwise. Although subcycles are eliminated using an extension of the formulation by Miller et al. [1960] for the Vehicle Routing Problem (VRP), the x^T_{iit} (where i is the depot) subcycle is allowed, representing an unused truck which stays at its home depot, not causing any additional costs. Decision variable r^T_{it} is used for the rank of node i and truck t for the subcycle elimination.[4]

Flow variables $x^V_{ijv} \in \{0, 1\} \quad \forall i, j \in \mathcal{A}^V; v \in \mathcal{V}$ and rank variables r^V_{iv} are defined in a similar way for the city freighters at the second level.

[4] cp. Miller et al. [1960]

Each vehicle is assigned to one satellite which is its home base. Let $s = h(v)$ be a function to express to which satellite s the vehicle v is assigned as its home base. If vehicle v is used to supply customers, the corresponding route starts and ends at satellite $h(v)$, serving customers in between. Otherwise $x^V_{iiv} = 1$ where $i = h(v)$, in other words the vehicle is not moved at all, thus not incurring any costs.

Binary indicating variable $y_{itv} \in \{0, 1\} \quad \forall i \in \mathcal{A}^V, t \in \mathcal{T}, v \in \mathcal{V}$ equals 1 if node i is served by truck t and vehicle v, directly or indirectly; 0 otherwise. This variable is needed for linking the first level of transportation $\mathcal{A}^T$ with the second level $\mathcal{A}^V$ and for synchronising the demanded quantities of goods. The "demand" of the satellites is not known from the given data of the problem set, as it depends on the customers served by the vehicles based at that specific satellite. Shifting one customer from a vehicle route originating at satellite i to a vehicle route originating at satellite j might violate the capacity constraint of the truck serving satellite j, so both levels have to be considered at the same time or iteratively.

4.2 Problem Formulation

The objective function

$$\min \text{ Total Cost} = \alpha \cdot \sum_{i,j \in \mathcal{A}^T} \sum_{t \in T} C_{ij} \cdot x^T_{ijt} + \sum_{i,j \in \mathcal{A}^V} \sum_{v \in V} C_{ij} \cdot x^V_{ijv} + \tag{1}$$

$$C_T \cdot \sum_{t \in \mathcal{T}} (1 - x^T_{00t}) + C_V \cdot \sum_{i \in \mathcal{S}} \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} y_{itv}$$

subject to the constraints

$$\sum_{j \in \mathcal{A}^T} x^T_{ijt} = \sum_{j \in \mathcal{A}^T} x^T_{jit} \qquad \forall\ i \in \mathcal{A}^T, t \in \mathcal{T} \tag{2}$$

$$\sum_{i \in \mathcal{A}^T} \sum_{t \in T} x^T_{ijt} \leq 1 \qquad \forall\ j \in \mathcal{S} \tag{3}$$

$$\sum_{j \in \mathcal{A}^T} x^T_{0jt} = 1 \qquad \forall\ t \in \mathcal{T} \tag{4}$$

$$\sum_{i \in \mathcal{A}^T} x^T_{ijt} \geq y_{jtv} \qquad \forall j \in \mathcal{S}, t \in \mathcal{T}, v \in \mathcal{V} \tag{5}$$

$$\sum_{t \in \mathcal{T}} y_{itv} = 1 - x^V_{iiv} \qquad \forall\ i \in \mathcal{S}, v \in \mathcal{V} \text{ where } i = h(v) \tag{6}$$

$$\sum_{i \in \mathcal{C}} \sum_{v \in \mathcal{V}} D_i \cdot y_{itv} \leq Q_T \qquad \forall t \in \mathcal{T} \tag{7}$$

$$y_{itv} + x^V_{ijv} \leq 1 + y_{jtv} \qquad \forall i, j \in \mathcal{A}^V, t \in \mathcal{T}, v \in \mathcal{V} \tag{8}$$

$$\sum_{j \in \mathcal{A}^V} x^V_{ijv} = \sum_{j \in \mathcal{A}^V} x^V_{jiv} \qquad \forall i \in \mathcal{A}^V, v \in \mathcal{V} \tag{9}$$

$$\sum_{i \in \mathcal{A}^V} \sum_{v \in \mathcal{V}} x^V_{ijv} = 1 \qquad \forall j \in \mathcal{C} \tag{10}$$

$$\sum_{j \in \mathcal{A}^V} x^V_{ijv} = 1 \qquad \forall i \in \mathcal{S}, v \in \mathcal{V} \text{ where } i = h(v) \tag{11}$$

$$\sum_{i \in \mathcal{A}^V : i \neq j} x^V_{ijv} = \sum_{t \in \mathcal{T}} y_{jtv} \qquad \forall j \in \mathcal{A}^V, v \in \mathcal{V} \tag{12}$$

$$\sum_{i \in \mathcal{C}} \sum_{t \in \mathcal{T}} D_i \cdot y_{itv} \leq Q_V \qquad \forall v \in \mathcal{V} \tag{13}$$

$$r^T_{it} + Q_T \cdot x^T_{ijt} \leq r^T_{jk} + Q_T - 1 \qquad \forall i, j \in \mathcal{S}, t \in \mathcal{T} \text{ where } i \neq j \tag{14}$$

$$\sum_{j \in \mathcal{C}} D_j \cdot y_{jtv} \leq r^T_{it} \qquad \forall i \in \mathcal{S}, t \in \mathcal{T}, v \in \mathcal{V} \tag{15}$$

$$r^T_{it} \leq Q_T \qquad \forall i \in \mathcal{S}, t \in \mathcal{T}, v \in \mathcal{V} \tag{16}$$

$$r^V_{iv} + Q_V \cdot x^V_{ijv} \leq r^V_{jv} + Q_V - D_j \qquad \forall i, j \in \mathcal{C}, v \in \mathcal{V} : i \neq j \wedge D_i + D_j \leq Q_v \tag{17}$$

$$D_i \leq r^V_{iv} \qquad \forall i \in \mathcal{C}, v \in \mathcal{V} \tag{18}$$

$$r^V_{iv} \leq Q_V \qquad \forall i \in \mathcal{C}, v \in \mathcal{V} \tag{19}$$

The objective function (1) simply calculates all accrued costs, thus summing up truck driving costs, vehicle driving costs, truck fixed costs and vehicle fixed costs.

Constraint (2) makes sure every node visited by a truck has to be left again. Constraint (3) states that every satellite is served at most once by a truck, as there are no split deliveries allowed. Constraint (4) forces every truck to "leave" the depot, where destination j may also be the depot, meaning the truck is not used at all, thus not incurring any fixed costs. Constraint (5) models with decision variable y which satellite is served by which truck. If a vehicle leaves from a satellite, the satellite has to be supplied by a truck, ensured with Constraint (6). Constraint (7) ensures that the truck capacity is not exceeded. Constraint (8) is used to backtrack which truck originally supplied the satellite from which a customer is served. This approach has been chosen to ensure linearity in the constraints.[5]

Constraint (9) works for the vehicles in the same way as Constraint (2) does for the trucks. Every node visited by a vehicle has to be left again. As every customer has to be served exactly once, (other than the satellites, where not all of them have to be used) the left side of Constraint (10) has to equal 1. The vehicles also have to "leave" their assigned home satellite, again also allowing a 1-node subcycle for unused vehicles not incurring any fixed costs. Constraint (12) ensures that decision variable y is set according to which customer is served by which city freighter and which truck had before delivered the goods to the satellite. Constraint (13) models that the capacity of the city freighters may not be exceeded.

[5]If we had used a 2-index formulation for y, i.e. $y^T_{i,t}$ for the first level (which truck visits which satellite) and $y^V_{j,v}$ for the second level, the calculation of the truck capacities would have looked like $y^T_{i,t} \cdot y^V_{j,v} \cdot D_j$.

We used an extension for the VRP of the formulation of Miller et al. [1960] to avoid subcycles. Constraints (14) to (16) use decision variable r^T to avoid subcycles at the truck level, according to the formulation of Miller et al. [1960]. Subcycle elimination for the second level is assured through Constraints (17) to (19), using decision variable r^V for the vehicles.

5 Metaheuristic Approach

5.1 Solution Statement

There are basically two ways to solve combinatorial optimisation problems such as e.g. the 2E-CVRP: by means of exact methods or with a (meta-) heuristic. As mentioned above the problem is NP-hard, so finding optimal or near-optimal solutions is limited to small sized instances. An effective metaheuristic can be quite fast, although as it is still only a heuristic it can not guarantee the optimality of the solution obtained.

Figure 5 illustrates the techniques used by a metaheuristic. It is a plot of search space on the horizontal axis over objective on the vertical axis. In the case of many routing problems the objective function calculates the costs. As we try to save costs we are dealing with a minimising problem: the lower the objective value gets, the better the found solution is. The search space axis shows the size of neighbourhoods between two solutions. If only small parts of a solution are changed, the objective also varies slightly. If we examine a totally different solution, it is more likely that we also obtain a completely different objective value. So the size of a neighbourhood is a kind of a measure for the similarity of two solutions.

The main goal of metaheuristics is a good combination of used neighbourhood operators. Local search algorithms like i.e. 2-opt or 3-opt are useful for exploring

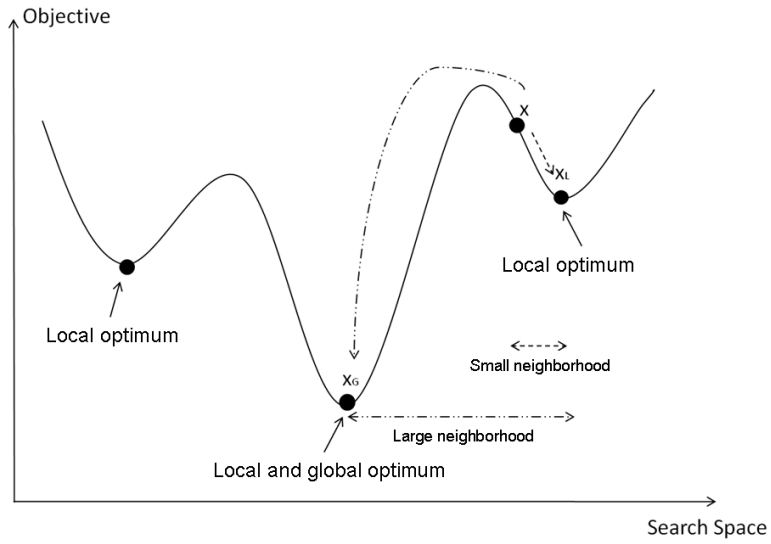

Figure 5: Exploring Search Space

small neighbourhoods and finding local optima. In order not to get stuck in one of these local optima the implementation of intelligent and wisely selected neighbourhood operators is crucial. As shown in Figure 5 the search space has to be explored far enough to find a new local minimum, ending up at the global optimum.

The used local search should be capable of finding the next local optimum (i.e. from x to x_L) fast. A metaheuristic has to make use of neighbourhoods large enough to pass by maxima to find other (local) minima, without turning the whole process into a "random walk". The step from x or x_L to x_G is a combination of a good (or, as there is often some randomness involved, *lucky*) neighbourhood operator and a local search afterwards.

Recent important new metaheuristics were the Variable Neighbourhood Search (VNS) by Mladenovic and Hansen [1997], Hansen and Mladenovic [2001] and

the Large Neighbourhood Search (LNS) by Schrimpf et al. [2000], Pisinger and Ropke [2010], on which the proposed metaheuristic is based.

There is no predefined assignment which customer has to be served from which satellite. The proposed heuristic solves the second level step first, to obtain "demands" for the satellites. As they don't serve as warehouses for goods, the exact amount that is shipped *from* a satellite has to be delivered *to* it by a truck before. For the now known amount of goods to ship to the satellites we can find a demand satisfying solution for the first level routing problem. These steps are then solved iteratively.

There exist several different metaheuristics to do so, which perform differently, depending on the application. The original goal was the development of a VNS similar to the work of Hansen and Mladenovic [2001].

5.2 Variable Neighbourhood Search

The VNS uses different nested neighbourhoods as they are needed. Starting with small neighbourhood steps (called "shaking"), the impact of the shaking steps is increased if no further improvement to another local optimum can be found. The main idea is the adoption of different shaking steps producing different neighbourhoods of solutions. A small neighbourhood step could be a single random move, so for example taking one customer from one route and moving it to another route. The different shaking operators used are examined in detail in Section 5.2.2. The steps of the basic VNS are shown in Algorithm 1 (cp. [Hemmelmayr et al., 2009, p. 793]).

Algorithm 1 Variable Neighbourhood Search

1: Initialisation: Select the neighbourhood structures $N_\kappa(\kappa = 1, \ldots, \kappa_{max})$ that will be used in the search; find an initial solution x; choose a stopping condition;
2: **repeat**
3: $\kappa \leftarrow 1$
4: **repeat**
5: Shaking: Generate a point x' at random from κ^{th} neighbourhood of $x(x' \in N_\kappa(x))$;
6: Local search: Apply some local search method with x' as initial solution; denote with x'' the so obtained local optimum;
7: **if** local optimum x'' is better than the incumbent x **then**
8: $x \leftarrow x'', N_1(\kappa \leftarrow 1)$
9: **else**
10: $\kappa \leftarrow \kappa + 1$
11: **end if**
12: **until** $\kappa = \kappa_{max}$
13: **until** stopping condition is met

5.2.1 Starting Solution

First of all we have to find any feasible starting solution, where all demands are satisfied and no constraints such as vehicle capacities, etc. are violated. This solution will then be improved step by step.

The starting solution is generated via *cheapest insertion*. We start with the second level, setting up city freighter routes from satellites to customers. For each customer all possible positions of insertion are calculated and the one with the lowest additional costs is chosen. So for the customer considered first this is basically the decision to supply him from the nearest satellite; the customer considered last can be inserted in any route at any position, where the city freighter still has enough capacity to deliver to that customer.

Figure 6 shows customer "x", who is not yet inserted in a route (left). Assuming the route shown still has enough capacity to deliver to "x", the search

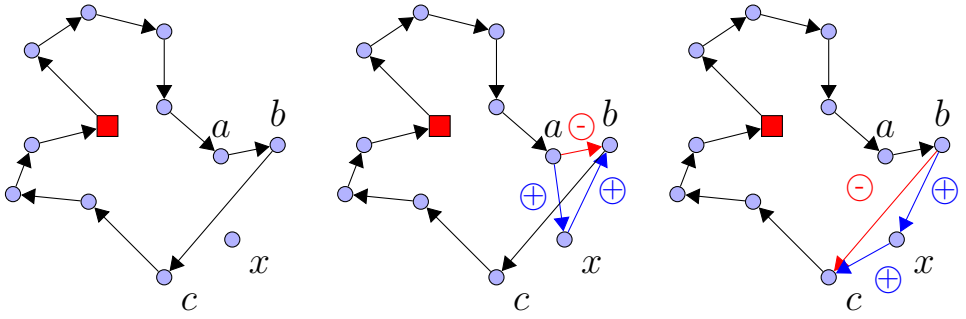

Figure 6: Cheapest Insertion

for the best position begins. The graph in the middle shows the additional cost for inserting "x" between customers "a" and "b", where the costs are higher than for inserting "x" between "b" and "c" (right). $C_{ax} + C_{xb} - C_{ab} > C_{bx} + C_{xc} - C_{bc}$

This is a greedy heuristic. The more customers are already inserted the worse the positions of the newly added customers might be. So a customer very near an existing route might have to be served by a city freighter coming from far away, simply because no other has still enough capacity left for that specific customer.

5.2.2 Shaking Steps

The smallest shaking operator is a single *move*. One customer is randomly selected. Different routes are also chosen randomly until one with sufficient free capacity on the vehicle to serve the customer is found. The customer then is inserted at any position on that route. In the case of the smallest step the next procedure would be a local search, which is described in Section 5.2.3. If no improvement was found after local search, the next larger shaking step is chosen.

Creating larger shaking steps can be achieved by either executing more single moves before performing a local search and testing the acceptance criterion or switching to other shaking operators. The next larger operator is the extension from a simple move to an *exchange*. Customers are exchanged between two routes,

which gives more freedom in terms of capacity. After one customer has been moved from the first to the second route, the latter can be temporarily overloaded. Now we search for an applicable customer which will be moved back from the second to the first route. There must be adequate demand so that none of both routes vehicles' capacities is exceeded afterwards. If a better solution is found after execution of local search, the neighbourhood size is reset to the smallest one and the search continues again, now starting from the better solution.

The shaking steps can be performed on both levels. Of course, satellites on truck routes can only be exchanged with satellites, and customers on city freighter routes only with other customers.

5.2.3 Local search

The 2-opt, first introduced by Croes [1958], was used as local search. It improves each vehicle tour independently from the others by relinking edges. We select and delete two edges from one tour, one from node i to node $i+1$ and a second one from j to $j+1$. These two edges are replaced by the relinking of node i to j and $i+1$ to $j+1$, as shown in Figure 7.

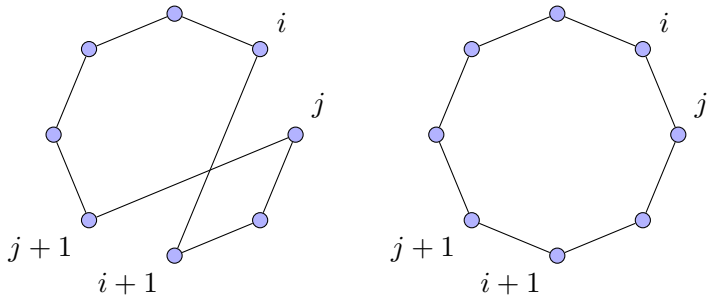

Figure 7: 2-Opt

Any improvement in route costs found by the 2-opt is accepted (2-opt first improvement). It is executed as long as improvements can be found, but no more than 100 times.

5.2.4 Acceptance and Termination Criterion

Any improvements in total costs after the shaking steps and local search are accepted and the new found solution then is the new incumbent solution. The termination was set after a fixed number of iterations.

An at least partial working VNS was implemented in C++. It was not very sophisticated and therefore very fast - but the results differed strongly from known solutions in literature. Most of the short time needed was wasted with searching possible customers which could be moved or exchanged. Often no move could be performed because of lack of capacity on the vehicles, so the starting solution was only slightly improved.

5.3 Large Neighbourhood Search

One problem which occurred very soon was that most of the capacity of city freighters has to be used. This is positive in terms of efficiency of our transportation goal, but absolutely bad for finding customers that can be moved to other routes. Customers with large demands could not be transferred to other routes at all, as there were no routes with sufficient capacity left. The exchange operator improves this problem just a little bit, as still only customers with similar demands can be exchanged in order not to overload a city freighter.

After facing some problems in debugging the C++ source code of the VNS implementation and long time of searching for possible memory leaks caused by the program not behaving as it should, we decided to re-implement the complete heuristic from scratch in Java. Memory management is a bit more comfortable

than in C++, even though the runtime might be negatively affected. At least we learned from the first attempt which functions and characteristics are more important to be considered from the very beginning of the implementation and data structure.

Shaw [1998] proposed a LNS where higher degrees of the solution are destroyed and repaired afterwards. LNS is based on the idea of "ruin and recreation", as it is called by Schrimpf et al. [2000], respectively "destroy and repair" by Pisinger and Ropke [2010]. The basic concept is the same; parts of the incumbent solution are completely destroyed, leading temporarily to a non-feasible solution, and then repaired afterwards. A defined number of nodes is deleted from the solution. The customers are then inserted in another way into the partial and infeasible solution, until all customers are served again. For further details see the Algorithm 2.[6]

Algorithm 2 Large Neighbourhood Search

```
1: INPUT: feasible starting solution x
2: x^b ← x;
3: repeat
4:    x^t ← r(d(x));
5:    if accept (x^t, x) then
6:       x ← x^t;
7:    end if
8:    if c(x^t) < c(x^b) then
9:       x^b ← x^t;
10:   end if
11: until stop criterion is met
12: return x^b
```

The LNS was recently enhanced by Pisinger and Ropke [2010], giving an overview of important parameter settings, especially the degree of destruction, leading them to a Very Large Neighbourhood Search (VLNS) and ALNS. They

[6] cp. [Pisinger and Ropke, 2010, p. 407]

focused mainly on the TSP and the CVRP. As the 2E-CVRP is mainly an extension of the CVRP, the use of their ideas was obvious: delete more nodes at the same time and insert them to the rest of the solution afterwards again, so that all the routes have some free headroom for the reallocation of nodes.

5.3.1 Starting Solution

The starting solution for the LNS was built the same way as for the VNS, described in Section 5.2.1.

5.3.2 Destroy & Repair

Most of the fine tuning and testing of different parameter settings is done at the destroy operator; therefore we will first introduce the repair technique, as it is similar to the generation of the starting solution just mentioned before.

Based on an already partially existing solution of the remaining, not-deleted nodes from the incumbent solution we use the same cheapest insertion algorithm as for the starting solution. Each customer who is not already served is inserted at the cheapest possible position at that moment.

The first attempt to destroy the solution was a random selection of customers that were deleted from the actual solution. The only parameter used was the percentage n of customers to be removed. Each destroyed solution will be repaired afterwards again to obtain only feasible solutions. This technique generated surprisingly good results given for its simplicity. To compensate for the greediness of the algorithm we wanted to bias the destruction more on eliminating the badly inserted customers. It is easy to perform a delta evaluation for each customer to get a measure of how good or bad this customer is at the actual position.

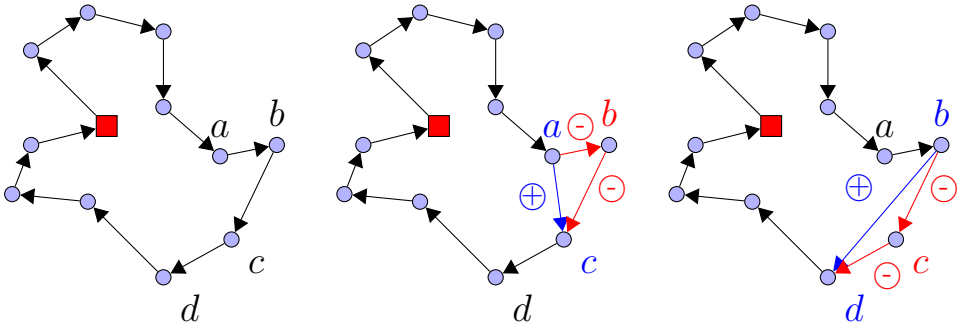

Figure 8: Delta Evaluation

Figure 8 shows the tour in the beginning (left), the calculation of savings for removing customer "b" (middle) and customer "c" (right). If customer "b" is no longer served by that vehicle, the changes of the tour cost ($\widehat{=}$ distance travelled) are calculated $C_{ac} - C_{ab} - C_{bc}$. In the case of removing customer "c" the calculation is analogously $C_{bd} - C_{bc} - C_{cd}$. Both will result in a decrease in costs, therefore the result has negative algebraic sign. The improvement for removing customer "b" is evidently higher than for removing customer "c".

The similarity to the repair method is obvious if you compare Figure 6 and Figure 8. After destruction of the solution many routes should have some excess capacity, which makes room for re-inserting the customers at better positions than they were before.

The measure of the delta evaluation cannot give absolute figures on the quality of a customer's position. The quality can only be compared to other customers' positions. Using the "roulette wheel" selection (see Figure 9) some randomness is added. The higher the savings for removing one customer, the more likely it will be selected for destruction.

The sequence of the deleted customers is also treated differently with a random operator: sometimes they are sorted for ascending demands, sometimes descending and sometimes they are chosen randomly for re-insertion.

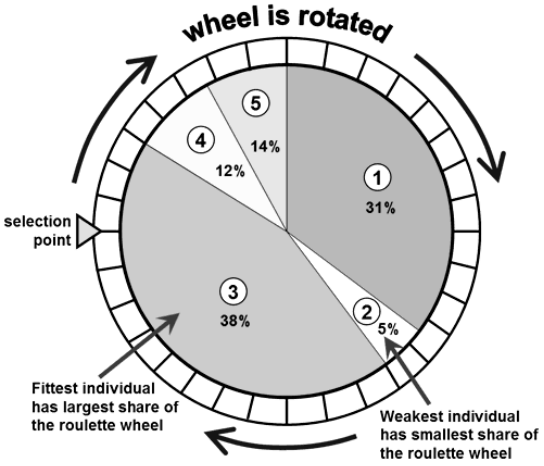

Figure 9: Biased Destroy Selection (Newcastle Engineering Design Centre, Merz Court, Newcastle University [2012])

Destroy and repair is performed only on the second level. The largest instances have only five satellites, the truck routes are always constructed only with cheapest insertion and local search.

5.3.3 Local Search

The local search used for the LNS is also a 2-opt, as described in Section 5.2.3. It was transformed to a "best improvement", i.e. all possible relinking paths are calculated and the one with the best improvement is chosen. The whole procedure is repeated until no more improvements can be found on that route, i.e. 2-optimal, but at most 100 times. Local search is also performed only on routes that had changed during the destroy and repair phase.

5.3.4 Acceptance and Termination Criterion

Like the VNS, the LNS also accepts only improvements (cp. Section 5.2.4). An important parameter setting was the number of iterations before the incumbent

solution was compared with the newly generated one. The termination criterion was set with a specific time limit. The parameter analysis for different iterations until comparison and runtime limits can be seen in Section 6.3.2.

6 Results

6.1 Instances

All the instances that were used for testing either the Mixed Integer Program (MIP) or the metaheuristic give locations for the nodes in a x-y coordinate system. The distances are calculated Euclidean. The instances used for testing and comparison to other publications are derived from Crainic et al. [2010][7]. They consist of a maximum of five satellites, so the hard part is the effective assignment and routing of the second level, which city freighters should deliver to which customers leaving from which satellite. With the "demands" of each satellite calculated afterwards the good routing of the trucks to the satellites is quite trivial and not too time-consuming, no matter which technique is used.

We used two different sets of instances for testing. The first one was self generated from the instances commonly referred to as the "Christofides instances"[8]. These were originally designed for the VRP, so we simply transformed the first n customers to satellites without a given demand and removed some of the customers, as we were not able to calculate optimal solutions for the original 50 nodes the instances consisted of. The specifications of these instances can be seen in Table 1.

They were mainly used to compare the solutions obtained by the MIP to solutions found by the metaheuristic.

[7] downloaded at http://people.brunel.ac.uk/~mastjjb/jeb/orlib/vrp2einfo.html
[8] cp. Christofides and Eilon [1969], Eilon et al. [1971], Christofides et al. [1979]

Table 1: Small Test Instances

Name	Satellites	Customers	Trucks	City Freighters
12n3s	3	8	2	3
13n3s	3	9	2	3
14n3s	3	10	2	3
15n3s	3	11	2	3

The other set of instances was derived from Crainic et al. [2010], with special focus on Instance50-37 to Instance50-54 from Set4, as these are solvable without split deliveries on the first level and many other publications present results for comparison. These Instances also have five potential Satellite locations, making the first level delivery a bit more interesting than with only two or three. The specifications of these instances are shown in Table 2.

Table 2: Set4 Test Instances

Inst.	Sat.	Cust.	Trucks	City Freighters	Sat. distrib.	Cust. distrib.
50-37	5	50	3	6	Random	Random
50-38	5	50	3	6	Random	Random
50-39	5	50	3	6	Sliced	Random
50-40	5	50	3	6	Sliced	Random
50-41	5	50	3	6	Forbidden	Random
50-42	5	50	3	6	Forbidden	Random
50-43	5	50	3	6	Random	Centroids
50-44	5	50	3	6	Random	Centroids
50-45	5	50	3	6	Sliced	Centroids
50-46	5	50	3	6	Sliced	Centroids
50-47	5	50	3	6	Forbidden	Centroids
50-48	5	50	3	6	Forbidden	Centroids
50-49	5	50	3	6	Random	Quadrants
50-50	5	50	3	6	Random	Quadrants
50-51	5	50	3	6	Sliced	Quadrants
50-52	5	50	3	6	Sliced	Quadrants
50-53	5	50	3	6	Forbidden	Quadrants
50-54	5	50	3	6	Forbidden	Quadrants

The different satellite distributions are as follows:

- *Random* distribution of satellites, without any further constraints. This might result in two satellites being located very close together, which is not too desirable for real transport patterns.
- A ring around the customers is divided into *slices*, where every satellite then is placed randomly in one slice.
- *Forbidden* zone represents a town e.g. next to the sea or a mountain where no satellites may be placed.

The different customer distributions are as follows:

- A *random* distribution of customers represents a regional distribution, where locations are randomly chosen with an uniform distribution.
- *Centroids* represent down town and suburb zones with higher density of customers.
- *Quadrants* simulate a town where large zones are separated by e.g. a river or a valley.

For further details on the creation and specifications of the instances we recommend [Crainic et al., 2010, p. 5948].

6.2 Solving the Mixed Integer Program

The described problem was implemented in Fico Xpress Optimization Suite[9] using Mosel language. For testing purposes we used the small instances described in Table 1. Instances up to 14 nodes could be solved to optimality with a total runtime limit of 4000 seconds.[10] The results are presented in Table 3.

[9] http://www.fico.com, 1.2.2012

[10] running on a Intel T7300 DualCore, 3 GB RAM on Win7 32bit

The largest instances which have been solved to optimality to our knowledge are by Perboli and Tadei [2010]. They presented optimal solutions for instances up to 33 customers and 2 satellites, after introducing many new inequalities used for branch-and-cut.

6.3 Results from Large Neighbourhood Search

6.3.1 Best Setting

The small instances, which have been solved by Xpress, were used to test the performance of the metaheuristic. Table 3 shows a comparison of runtimes (in seconds) for the exact method versus the herein described LNS. The solutions found by the metaheuristic are always the same as the ones Xpress found, but within times in different orders of magnitude. The instance with 15 nodes still had a gap of more than 28% after 4000 seconds of Xpress runtime. Column "AvgSol." shows the average solution of seven runs, "BestSol." shows the best solution found within that seven runs. "TotalT." represents the overall runtime of the LNS, whereas "SolT." shows the time when the final solution was first found.

Table 3: Comparison Xpress - Large Neighbourhood Search

	Xpress			LNS			
Inst.	Sol.	T.	Gap	AvgSol.	BestSol.	TotalT.	SolT.
10n3s	175.76	3.7	0.00%	**175.76**	**175.76**	0.3	0.1
11n3s	252.40	39.4	0.00%	**252.40**	**252.40**	0.4	0.2
12n3s	253.19	413.2	0.00%	**253.19**	**253.19**	1.4	0.8
13n3s	255.54	600.6	0.01%	**255.54**	**255.54**	0.7	0.1
14n3s	286.66	2516.0	0.01%	**286.66**	**286.66**	1.1	0.2
15n3s	315.77	4000.0	28.15%	**315.77**	**315.77**	0.9	0.2

The results for the set 4 instances obtained by the LNS are presented in Table 4. We show the mean of seven runs for each instance as well as the best found solutions. Column "Gap" represents the difference of the average Solutions compared to the best known solutions in percent. The total runtime was limited to 30 seconds, column "SolT." shows the mean time in seconds until the solution was found.

Table 4: Results from Large Neighbourhood Search

Instance	BKS	AvgSol	Gap	BestSol	SolTime
50-37	1545.99	1554.09	0.52%	**1528.98**	9.6
50-38	**1172.83**	1225.33	4.48%	1218.19	7.3
50-39	1525.24	**1522.72**	-0.17%	**1522.28**	6.5
50-40	**1197.00**	1200.20	0.27%	1200.20	7.4
50-41	1687.96	**1664.08**	-1.41%	**1663.01**	23.8
50-42	**1191.46**	1228.31	3.09%	1192.48	16.1
50-43	1453.11	1506.45	3.67%	**1445.75**	12.5
50-44	**1047.96**	1115.67	6.46%	1112.99	8.2
50-45	1480.32	**1475.20**	-0.35%	**1456.98**	18.3
50-46	**1074.88**	1111.09	3.37%	1111.09	8.0
50-47	1620.70	**1603.50**	-1.06%	**1587.73**	16.4
50-48	**1078.28**	1084.91	0.62%	1078.57	10.3
50-49	1499.52	**1490.36**	-0.61%	**1446.73**	21.4
50-50	**1072.42**	1124.74	4.88%	1124.74	5.1
50-51	1435.83	**1399.72**	-2.51%	**1399.72**	13.0
50-52	1132.20	**1128.64**	-0.31%	**1128.64**	5.0
50-53	1569.59	1570.24	0.04%	**1569.54**	11.8
50-54	1189.14	**1147.55**	-3.50%	**1115.23**	5.1
Mean	1331.91	1341.82	0.74%	**1327.94**	11.4

The best known solutions for the instances used for comparison are derived from [Mancini et al., 2011, p. 27].

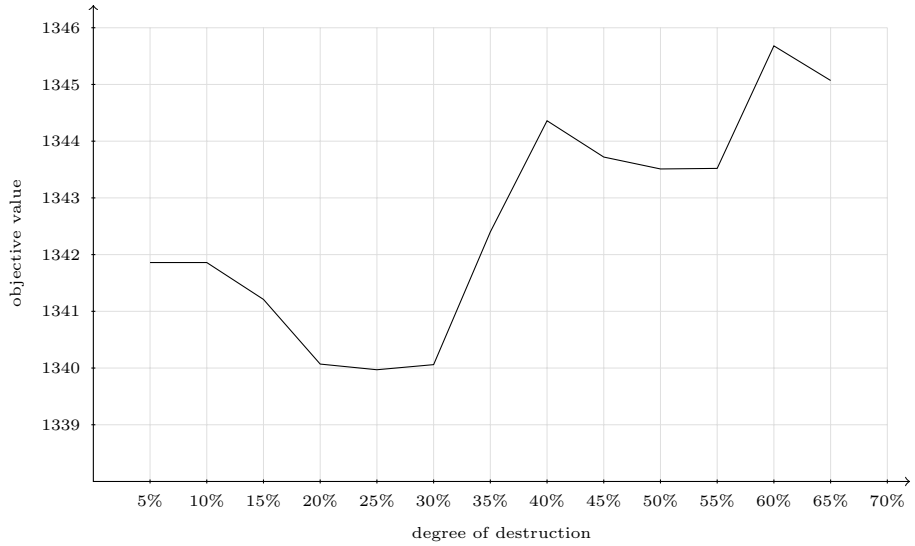

Figure 10: Average solutions for different percentages of destruction

6.3.2 Parameter Test

There are many different parameters which needed fine tuning as, for instance, the destroy operator. Figure 10 shows the average objective value of five runs for each of the instances 50-37 to 50-54 and the relation to different percentages of destruction. If on the one hand only 5% of the incumbent solution are destroyed, the impact is not sufficient for exploring the search space far enough to find better solutions. On the other hand for too high levels of destruction the solution does not converge towards better solutions, as it is simply disturbed too much in each iteration. The optimum is about 25% of destruction.

Another important factor for the performance of the metaheuristic is the runtime. Obviously we can show that the longer the runtime is, the better the results are. As shown in Figure 11 after approximately 30 seconds of runtime there are no great achievements any more. Longer runtimes might result in not significant but single lucky good solutions. The graph (Figure 11) again shows the mean over five runs of instances 50-37 to 50-54.

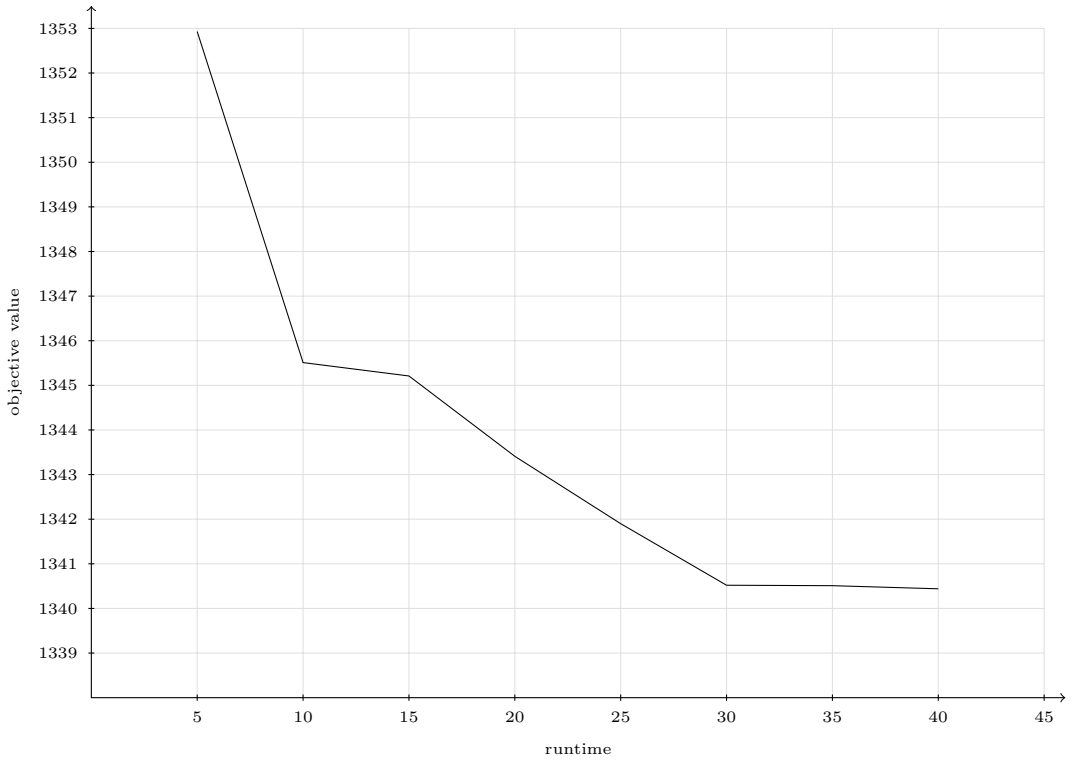

Figure 11: Average solutions for different runtimes

7 Conclusion

We showed a working LNS for the 2E-CVRP that performs well. We found new best known solutions for more than half of the test instances from set 4 that are solvable without split deliveries. The metaheuristic found the optimal solutions for all the small instances that our exact implementation could solve to optimality, within fractions of the computing time needed by the MIP.

Further research will have to focus on split deliveries, as e.g. Hemmelmayr et al. [2011] did. Split deliveries of the satellites are closer to real-world circumstances and will boost the efficiency of satisfying customer demands in terms of transport.

Appendix

We provide detailed results for new found best solutions here.

```
Instance50-37.dat
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 14205.63 Load: 9703 Depot: 0 [1]
Truck 2 cap: 12500 Cost: 24906.22 Load: 9651 Depot: 0 [3]
Truck 3 cap: 12500 Cost: 20327.32 Load: 8799 Depot: 0 [5]
VehicleRouteList CF:
CF1a  cap: 5000 Cost: 15796.59 Load: 4886 Depot: 1 [16 22 10 6 34 28 18 44]
CF1b  cap: 5000 Cost: 13139.72 Load: 4817 Depot: 1 [20 23 19 48 17 38 37 24]
CF3a  cap: 5000 Cost: 10077.95 Load: 4799 Depot: 3 [46 9 43 40 21 31 7]
CF3b  cap: 5000 Cost: 20091.88 Load: 4852 Depot: 3 [35 8 54 25 15 42 14 52 53 30 33]
CF5a  cap: 5000 Cost: 13992.11 Load: 3994 Depot: 5 [49 26 47 50 55 27 36 45 39]
CF5b  cap: 5000 Cost: 20360.74 Load: 4805 Depot: 5 [12 32 51 13 11 41 29]
Total Cost: 152898.17
```

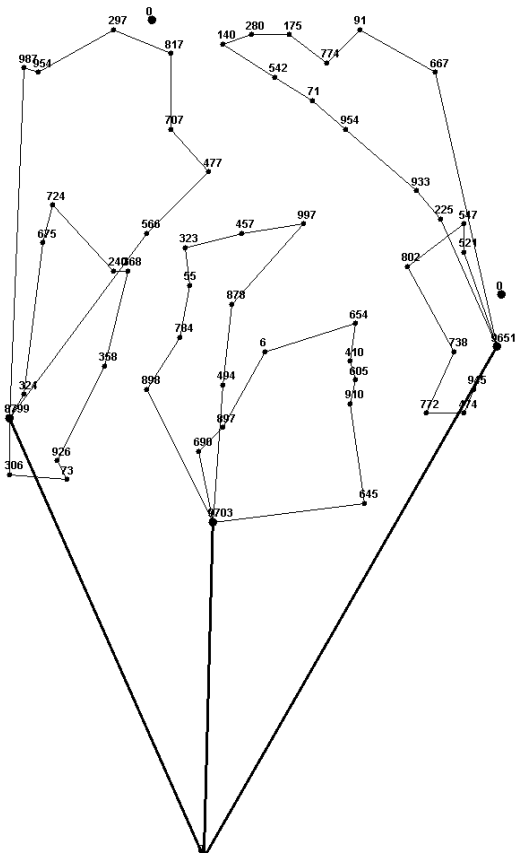

Figure 12: Instance50-37

```
Instance50-39.dat
```

VehicleRouteList Trucks:

Truck 1 cap: 12500 Cost: 222.04 Load: 9181 Depot: 0 [1]

Truck 2 cap: 12500 Cost: 146.87 Load: 9400 Depot: 0 [2]

Truck 3 cap: 12500 Cost: 323.54 Load: 9572 Depot: 0 [3]

VehicleRouteList CF:

CF1a cap: 5000 Cost: 98.35 Load: 4711 Depot: 1 [46 7 31 35 8 21 40]

CF1b cap: 5000 Cost: 99.88 Load: 4470 Depot: 1 [48 17 38 37 24 43 9]

CF2a cap: 5000 Cost: 122.36 Load: 4472 Depot: 2 [26 47 49 39 50 44 23 20]

CF2b cap: 5000 Cost: 180.48 Load: 4928 Depot: 2 [16 19 22 10 6 34 29 55 28 18]

CF3a cap: 5000 Cost: 129.37 Load: 4764 Depot: 3 [12 32 11 41 27 45 36]

CF3b cap: 5000 Cost: 199.38 Load: 4808 Depot: 3 [15 25 54 33 30 53 52 14 42 13 51]

Total Cost: 1522.28

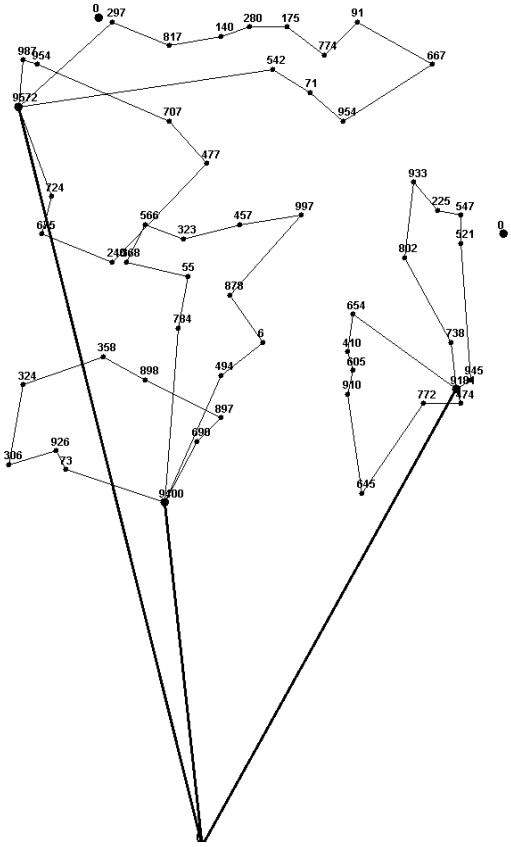

Figure 13: Instance50-39

Instance50-41.dat

VehicleRouteList Trucks:

Truck 1 cap: 12500 Cost: 354.68 Load: 9332 Depot: 0 [1]

Truck 2 cap: 12500 Cost: 206.71 Load: 9640 Depot: 0 [3]

Truck 3 cap: 12500 Cost: 229.48 Load: 9181 Depot: 0 [5]

VehicleRouteList CF:

```
CF1a  cap: 5000 Cost: 140.14 Load: 4511 Depot: 1 [42 14 52 53 30 33 54 25 15 13]
CF1b  cap: 5000 Cost: 141.09 Load: 4821 Depot: 1 [11 41 45 36 12 32 51]
CF3a  cap: 5000 Cost: 203.11 Load: 4966 Depot: 3 [23 20 26 47 49 39 50 44 16]
CF3b  cap: 5000 Cost: 188.17 Load: 4674 Depot: 3 [19 22 18 28 55 27 29 34 6 10]
CF5a  cap: 5000 Cost:  94.72 Load: 4711 Depot: 5 [46 40 21 8 35 31 7]
CF5b  cap: 5000 Cost: 104.88 Load: 4470 Depot: 5 [9 43 24 37 38 17 48]
Total Cost: 1663.01
```

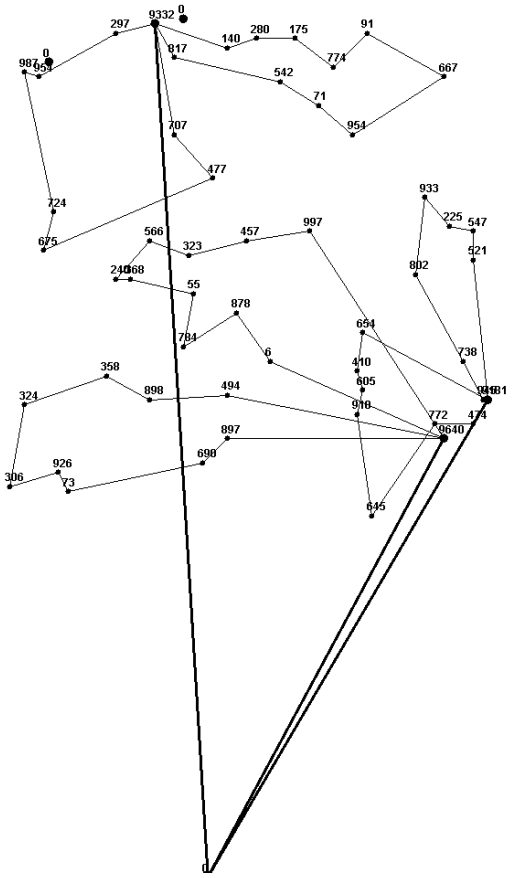

Figure 14: Instance50-41

```
Instance50-43.dat
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 14205.63 Load: 8281 Depot: 0 [1]
Truck 2 cap: 12500 Cost: 26929.54 Load: 9902 Depot: 0 [4]
Truck 3 cap: 12500 Cost: 20327.32 Load: 9970 Depot: 0 [5]
VehicleRouteList CF:
CF1a  cap: 5000 Cost:  9666.12 Load: 3301 Depot: 1 [37 38 40 36 39]
CF1b  cap: 5000 Cost: 12553.42 Load: 4980 Depot: 1 [19 18 16 9 22 21 23 24]
CF4a  cap: 5000 Cost: 12466.81 Load: 4923 Depot: 4 [34 25 7 6 10 8 32 33]
CF4b  cap: 5000 Cost: 18569.16 Load: 4979 Depot: 4 [35 31 15 13 14 55 51 52 53 54]
```

CF5a cap: 5000 Cost: 16821.34 Load: 4974 Depot: 5 [30 11 12 48 46 50 47 49]

CF5b cap: 5000 Cost: 13036.13 Load: 4996 Depot: 5 [45 44 41 43 20 17 26 28 29 27 42]

Total Cost: 144575.48

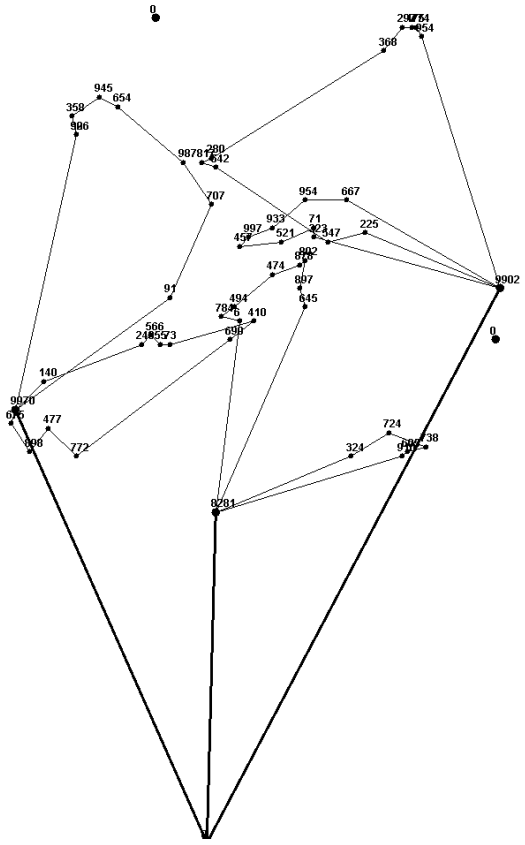

Figure 15: Instance50-43

Instance50-45.dat

VehicleRouteList Trucks:

Truck 1 cap: 12500 Cost: 22203.60 Load: 8285 Depot: 0 [1]

Truck 2 cap: 12500 Cost: 14687.41 Load: 9939 Depot: 0 [2]

Truck 3 cap: 12500 Cost: 32354.29 Load: 9929 Depot: 0 [3]

VehicleRouteList CF:

CF1a cap: 5000 Cost: 5735.66 Load: 3301 Depot: 1 [36 39 37 38 40]

CF1b cap: 5000 Cost: 11403.19 Load: 4984 Depot: 1 [24 23 22 21 34 31 33 35]

CF2a cap: 5000 Cost: 15165.90 Load: 4999 Depot: 2 [27 30 11 6 10 8 9 17 20]

CF2b cap: 5000 Cost: 14685.32 Load: 4940 Depot: 2 [43 41 44 45 42 29 28 26 18 16 19]

CF3a cap: 5000 Cost: 8321.92 Load: 4993 Depot: 3 [47 49 12 13 48 46 50]

CF3b cap: 5000 Cost: 21140.83 Load: 4936 Depot: 3 [14 15 7 25 32 54 53 52 51 55]

Total Cost: 145698.12

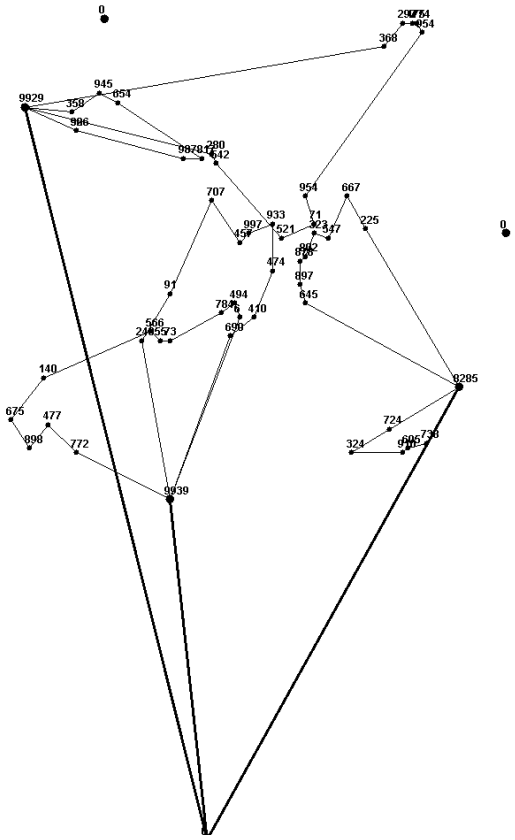

Figure 16: Instance50-45

```
Instance50-47.dat
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 206.71 Load: 8258 Depot: 0 [3]
Truck 2 cap: 12500 Cost: 344.38 Load: 9967 Depot: 0 [4]
Truck 3 cap: 12500 Cost: 229.48 Load: 9928 Depot: 0 [5]
VehicleRouteList CF:
CF3a  cap: 5000 Cost: 132.84 Load: 4957 Depot: 3 [17 19 20 18 16 6 10 9 24]
CF3b  cap: 5000 Cost:  42.74 Load: 3301 Depot: 3 [40 38 37 39 36]
CF4a  cap: 5000 Cost:  85.05 Load: 4993 Depot: 4 [50 47 49 12 13 48 46]
CF4b  cap: 5000 Cost: 221.40 Load: 4974 Depot: 4 [42 45 44 41 43 27 29 28 26 30 11 14]
CF5a  cap: 5000 Cost: 207.75 Load: 4956 Depot: 5 [35 33 32 15 55 51 52 53 54]
CF5b  cap: 5000 Cost: 117.39 Load: 4972 Depot: 5 [31 34 25 8 7 21 22 23]
Total Cost: 1587.73
```

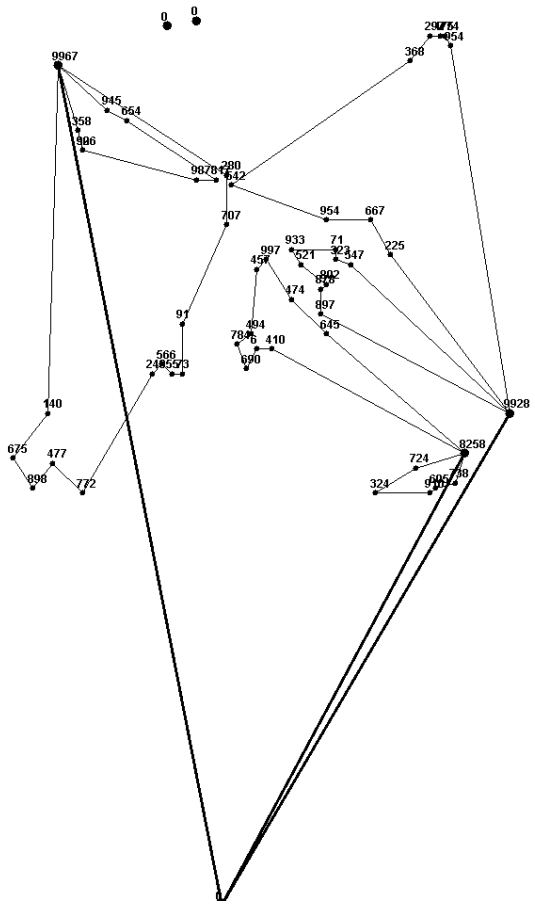

Figure 17: Instance50-47

```
Instance50-49.dat
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 142.06 Load: 9688 Depot: 0 [1]
Truck 2 cap: 12500 Cost: 270.22 Load: 9671 Depot: 0 [3 4]
Truck 3 cap: 12500 Cost: 203.27 Load: 8794 Depot: 0 [5]
VehicleRouteList CF:
CF1a  cap: 5000 Cost: 196.98 Load: 4910 Depot: 1 [49 47 50 55 45 40 38 43 30 25]
CF1b  cap: 5000 Cost: 128.40 Load: 4778 Depot: 1 [16 18 6 11 12 31 21]
CF3a  cap: 5000 Cost: 101.85 Load: 4974 Depot: 3 [8 15 14 13 10 17 9 7]
CF4a  cap: 5000 Cost: 123.57 Load: 4697 Depot: 4 [54 53 52 51 44 48 46]
CF5a  cap: 5000 Cost:  77.16 Load: 4050 Depot: 5 [27 29 28 19 22 23 20 24 26]
CF5b  cap: 5000 Cost: 203.22 Load: 4744 Depot: 5 [39 36 34 32 41 42 33 35 37]
Total Cost: 1446.73
```

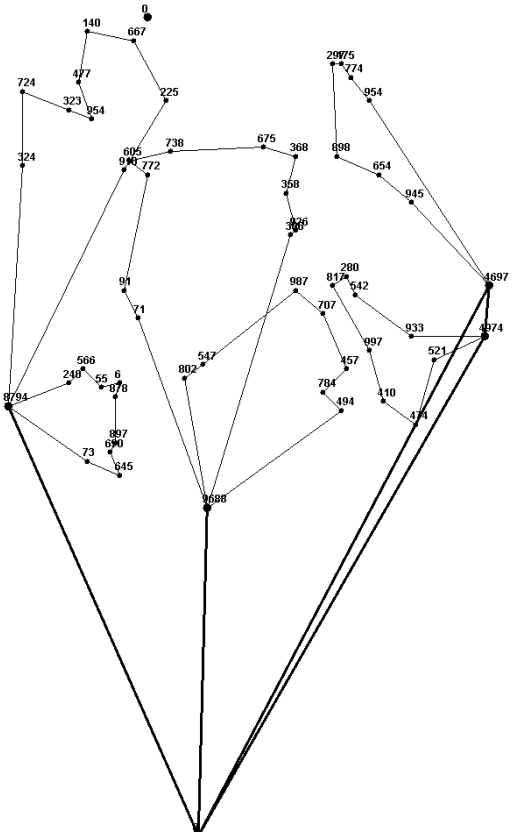

Figure 18: Instance50-49

```
Instance50-51.dat
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 222.04 Load: 9525 Depot: 0 [1]
Truck 2 cap: 12500 Cost: 146.87 Load: 8894 Depot: 0 [2]
Truck 3 cap: 12500 Cost: 323.54 Load: 9734 Depot: 0 [3]
VehicleRouteList CF:
CF1a  cap: 5000 Cost:  70.67 Load: 4549 Depot: 1 [9 17 16 18 6 10 8]
CF1b  cap: 5000 Cost: 134.43 Load: 4976 Depot: 1 [7 46 48 44 55 50 47 49]
CF2a  cap: 5000 Cost: 168.75 Load: 4850 Depot: 2 [21 31 11 15 14 13 12 30 25 19]
CF2b  cap: 5000 Cost:  75.98 Load: 4044 Depot: 2 [20 23 22 28 29 27 26 24]
CF3a  cap: 5000 Cost:  81.04 Load: 4766 Depot: 3 [39 37 43 38 32 41 36]
CF3b  cap: 5000 Cost: 176.40 Load: 4968 Depot: 3 [34 40 45 54 53 52 51 35 33 42]
Total Cost: 1399.72
```

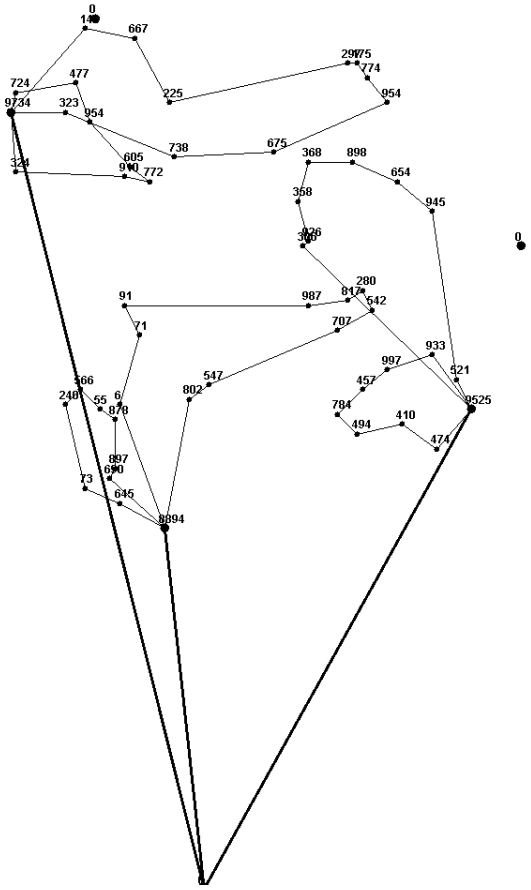

Figure 19: Instance50-51

Instance50-52.dat

VehicleRouteList Trucks:

Truck 1 cap: 12500 Cost: 175.97 Load: 9517 Depot: 0 [1 2]

Truck 2 cap: 12500 Cost: 397.51 Load: 10689 Depot: 0 [3 5]

VehicleRouteList CF:

CF1a cap: 5000 Cost: 129.88 Load: 4875 Depot: 1 [15 7 10 8 18 12 9 13 17 6 11 14]

CF2a cap: 5000 Cost: 170.82 Load: 4642 Depot: 2 [30 16 27 22 20 26 28 29 23 24 31 21 19 25]

CF3a cap: 5000 Cost: 103.60 Load: 4949 Depot: 3 [43 32 33 34 36 41 42 40 37 38 35]

CF3b cap: 5000 Cost: 12.00 Load: 61 Depot: 3 [39]

CF5a cap: 5000 Cost: 34.91 Load: 1029 Depot: 5 [55 45]

CF5b cap: 5000 Cost: 103.96 Load: 4650 Depot: 5 [47 50 48 44 54 46 52 53 49 51]

Total Cost: 1128.64

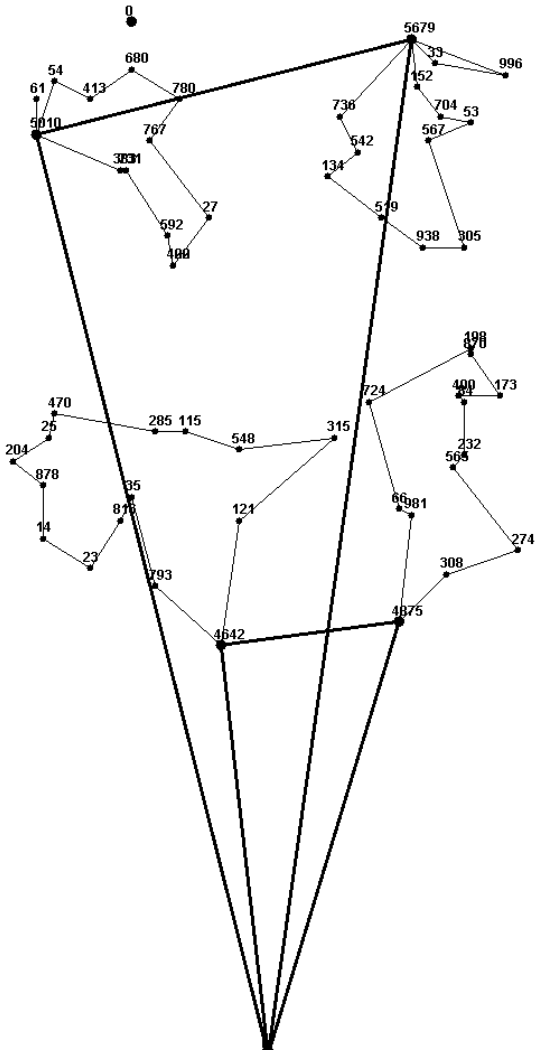

Figure 20: Instance50-52

```
Instance50-53.dat
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 206.71 Load: 9892 Depot: 0 [3]
Truck 2 cap: 12500 Cost: 344.38 Load: 9734 Depot: 0 [4]
Truck 3 cap: 12500 Cost: 229.48 Load: 8527 Depot: 0 [5]
VehicleRouteList CF:
CF3a  cap: 5000 Cost: 202.69 Load: 4996 Depot: 3 [24 26 20 23 22 19 28 27 29 30 25 18]
CF3b  cap: 5000 Cost: 132.89 Load: 4896 Depot: 3 [16 21 31 12 11 13 15]
CF4a  cap: 5000 Cost:  87.36 Load: 4766 Depot: 4 [36 39 37 43 38 32 41]
CF4b  cap: 5000 Cost: 166.02 Load: 4968 Depot: 4 [42 33 35 51 52 53 54 45 40 34]
CF5a  cap: 5000 Cost: 134.88 Load: 4735 Depot: 5 [14 49 47 50 55 44 48 46]
CF5b  cap: 5000 Cost:  65.12 Load: 3792 Depot: 5 [7 8 10 6 17 9]
Total Cost: 1569.54
```

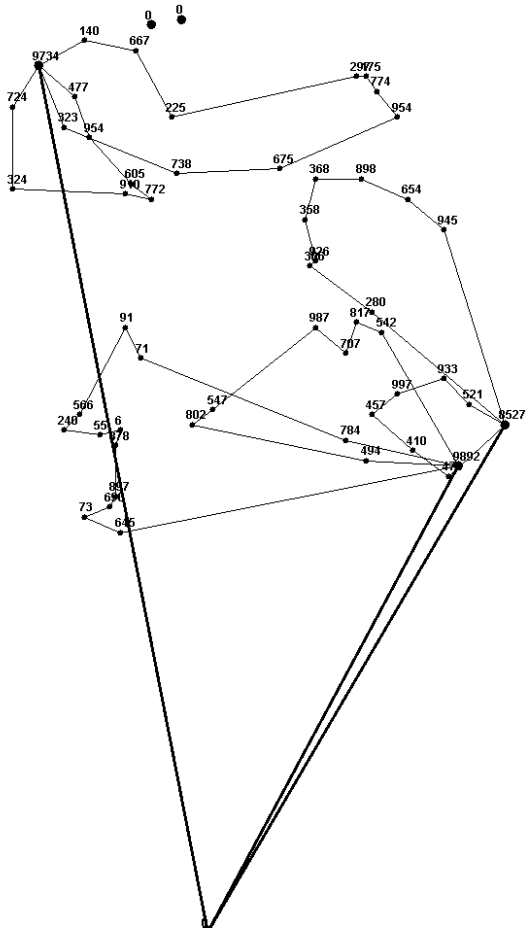

Figure 21: Instance50-53

```
Instance50-54
VehicleRouteList Trucks:
Truck 1 cap: 12500 Cost: 19416.49 Load: 10896 Depot: 0 [3]
Truck 2 cap: 12500 Cost: 22953.87 Load: 9310 Depot: 0 [5]
VehicleRouteList CF:
CF3a  cap: 5000 Cost: 23097.58 Load: 4803 Depot: 3 [7 16 41 53 49 51 47 45 55 48 50 44 54 12]
CF3b  cap: 5000 Cost: 12508.43 Load: 4530 Depot: 3 [17 13 9 18 8 46 52 10 6]
CF3c  cap: 5000 Cost:  4473.60 Load: 1563 Depot: 3 [15 14 11]
CF5a  cap: 5000 Cost: 15781.72 Load: 4983 Depot: 5 [34 36 33 43 32 42 40 37 38 35 39]
CF5b  cap: 5000 Cost: 13291.55 Load: 4327 Depot: 5 [28 26 20 22 27 30 19 21 25 31 24 23 29]
Total Cost: 111523.23
```

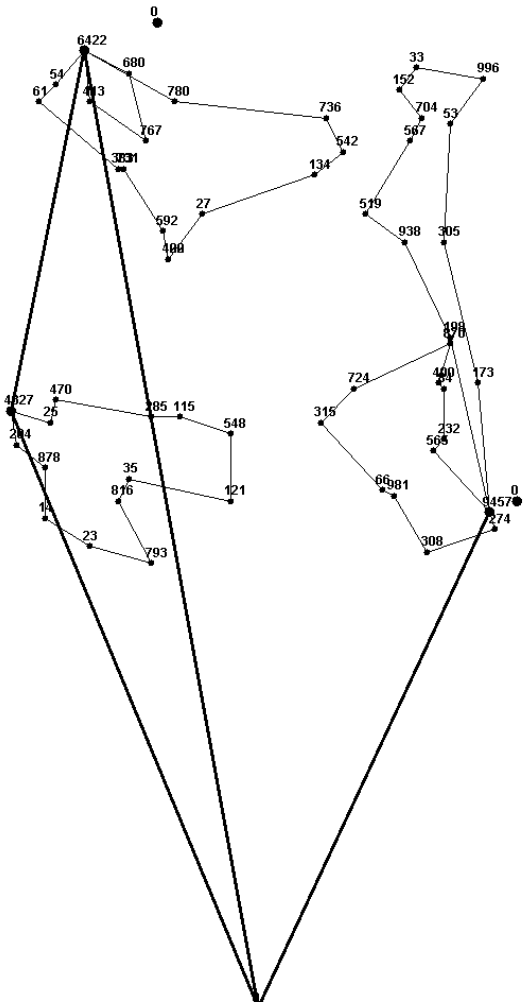

Figure 22: Instance50-54

List of Abbreviations

2E-CVRP	Two-Echelon Capacitated Vehicle Routing Problem
ALNS	Adaptive Large Neighbourhood Search
cp.	compare
Cust.	Customer
CVRP	Capacitated Vehicle Routing Problem
distrib.	distribution
e.g.	example given
etc.	et cetera
i.e.	id est
Inst.	Instance
LNS	Large Neighbourhood Search
MD-VRP	Multi-Depot Vehicle Routing Problem
MIP	Mixed Integer Program
NP	non-deterministic polynomial-time
p.	page
Sat.	Satellite
Sol.	Solution
T.	Time
TSP	Travelling Salesman Problem
VLNS	Very Large Neighbourhood Search
VNS	Variable Neighbourhood Search
VRP	Vehicle Routing Problem

List of Figures

List of Tables

List of Algorithms

References

D. Applegate, R. Bixby, V. Chvátal, and W. Cook. Milestones in the solution of tsp instances. Retreived online 1.2.2012, 4 2001. URL `http://www.tsp.gatech.edu/d15sol/index.html`.

D. Applegate, R. Bixby, V. Chvátal, and W. Cook. *The traveling salesman problem: a computational study*. Princeton University Press, Princeton, 2006. ISBN 0691129932.

R. Baldacci, A. Mingozzi, R. Roberti, and R. W. Calvo. An exact method for the two-echelon capacitated vehicle routing problem. Presentation at ROUTE 2011, June 3rd, Sitges, Spain; under revision for Operations Research, 2011.

N. Christofides and S. Eilon. An algorithm for the vehicle-dispatching problem. *OR*, pages 309–318, 1969.

N. Christofides, A. Mingozzi, P. Toth, and C. Sandi. Combinatorial optimization. In *A Wiley-Interscience Publication, Based on a series of lectures, given at the Summer School in Combinatorial Optimization, held in Sogesta, Italy, May 30th-June 11th, 1977, Chichester: Wiley, 1979, edited by Christofides, Nicos*, volume 1, 1979.

T. Crainic, N. Ricciardi, and G. Storchi. Models for evaluating and planning city logistic transportation systems. *Transportation science*, 43, 2007.

T. Crainic, S. Mancini, G. Perboli, and R. Tadei. Clustering-based heuristics for the two-echelon vehicle routing problem. Technical report, CIRRELT-2008-46 november report, 2008.

T. Crainic, G. Perboli, S. Mancini, and R. Tadei. Two-echelon vehicle routing problem: a satellite location analysis. *Procedia-Social and Behavioral Sciences*, 2(3):5944–5955, 2010. ISSN 1877-0428.

T. G. Crainic. City logistics. *2008 Tutorials in Operations Research: State-of-the-Art Decision-Making Tools in the Information-Intensive Age*, 10.1287:181–212, 2008.

T. G. Crainic, N. Ricciardi, and G. Storchi. Advanced freight transportation systems for congested urban areas. *Transportation Research Part C: Emerging Technologies*, 12(2):119–137, 2004. ISSN 0968-090X.

T. G. Crainic, N. Ricciardi, and G. Storchi. Models for evaluating and planning city logistics systems. *Transportation science*, 43(4):432–454, 2009. ISSN 1526-5447.

G. Croes. A method for solving traveling-salesman problems. *Operations Research*, pages 791–812, 1958.

H. Crowder and M. Padberg. Solving large-scale symmetric travelling salesman problems to optimality. *Management Science*, pages 495–509, 1980.

G. Dantzig and J. Ramser. The truck dispatching problem. *Management science*, pages 80–91, 1959.

G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, pages 393–410, 1954.

S. Eilon, C. Watson-Gandy, and N. Christofides. *Distribution management: mathematical modelling and practical analysis*. Griffin, 1971.

J. Feliu, G. Perboli, R. Tadei, and D. Vigo. The two-echelon capacitated vehicle routing problem. Technical report, Technical Report OR/02/08, Politecnico di Torino, 2008.

S. Gragnani, G. Valenti, and M. Valentini. City logistics in italy: A national project. In *Logistics Systems for Sustainable Cities, proceedings of the 3rd international conference on City Logistics*, pages 279–294, 2004.

P. Hansen and N. Mladenovic. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467, 2001. ISSN 0377-2217.

V. Hemmelmayr, K. Doerner, and R. Hartl. A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195(3):791–802, 2009.

V. Hemmelmayr, J. Cordeau, and T. Crainic. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Technical report, Technical Report CIRRELT-2011-42, Université de Montréal, 2011.

S. Mancini, T. G. Crainic, G. Perboli, and R. Tadei. A grasp with path-relinking metaheuristic for the two-echelon vehicle routing problem. Presentation at 9th Metaheuristics International Conference, July 27th 2011, Udine, Italy, July 2011.

K. Menger. Das botenproblem. *Ergebnisse eines mathematischen kolloquiums*, 2: 11–12, 1932.

C. Miller, A. Tucker, and R. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7 (4):326–329, 1960.

N. Mladenovic and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997. ISSN 0305-0548.

Newcastle Engineering Design Centre, Merz Court, Newcastle University, January 18th 2012. URL `http://www.edc.ncl.ac.uk/assets/hilite_graphics/rhjan07g02.png`.

Organisation for Economic Co-operation and Development (OECD). Delivering the gods: 21st century challenges to urban goods transport. Technical report, OECD Publishing, 2003. URL `http://www.oecdbookshop.org`.

M. Padberg and G. Rinaldi. Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters*, 6(1):1–7, 1987.

G. Perboli and R. Tadei. New families of valid inequalities for the two-echelon vehicle routing problem. *Electronic Notes in Discrete Mathematics*, 36:639–646, 2010.

G. Perboli, R. Tadei, and D. Vigo. The Two-Echelon Capacitated Vehicle Routing Problem: Models and Math-Based Heuristics. *Publication CIRRELT*, 55:1–42, 2008.

D. Pisinger and S. Ropke. Large neighborhood search. *Handbook of Metaheuristics*, pages 399–419, 2010.

G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171, 2000.

P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. *Principles and Practice of Constraint Programming—CP98*, pages 417–431, 1998.

Deutsche Zusammenfassung

Verschiedene Konzepte der „City Logistic“ behandeln die Auslieferung von Waren in dicht besiedelten Ballungsräumen. Einige davon laufen auf eine zweistufige Belieferung der Endkunden hinaus. Die Waren werden dabei in Zwischenlagern, sogenannten Satelliten, von großen Lastwägen auf kleinere Lieferwägen umgeladen. Das grundlegende Prinzip stammt ursprünglich aus Rom, wo große Lastwägen nicht durch die engen Straßen im historischen Stadtkern passen. Die Satelliten liegen meist mit günstiger Verkehrsanbindung am Stadtrand verteilt.

Wir zeigen ein mathematisches Modell für dieses „2E-CVRP“ (zweistufiges kapazitiertes Vehikel Routing Problem) und optimale Lösungen für kleine Testbeispiele, für die das Problem in annehmbarer Zeit exakt lösbar ist. Weiters zeigen wir die Implementierung einer Metaheuristik mit lokaler Suche, die auch für größere Aufgabenstellungen in relativ kurzer Zeit gute Ergebnisse liefert. Die optimalen Lösungen der kleinen Testbeispiele werden von der Metaheuristik in Bruchteilen der Rechenzeit gefunden, die das exakte Verfahren dafür braucht. Für mehr als die Hälfte der getesteten bekannten Problemstellungen wurden die bisher besten bekannten Lösungen verbessert.

Curriculum Vitae

Personal Data

First- and Surname: Ulrich Breunig

Date of Birth: March 22, 1982

Nationality: Austria

Education

2008	Erasmus scholarship at the "Universitat de Valencia", Valencia, Spain
2004 - 2012	Enrolled at the University of Vienna for International Business Administration
2002 - 2004	Study Program for "Electrical Engineering and Information Technology specialising in Computer Network Engineering" at the TGM Vienna with final year project: "Mobile IP – Roaming in PC-Networks", which was bought afterwards by the Austrian Railway Company (ÖBB)
2000 - 2001	Enrolled at the Technical University of Vienna for Electrical Engineering
1992 - 2000	Secondary school Schottengymnasium, Vienna
1988 - 1992	Primary school Judenplatz, Vienna

Working Experience

Feb. 2009 - Oct. 2011	Teaching Assistant at the Chair of Production and Operations Management, University of Vienna, Austria
since 2007	Self-employed audio and light engineer for event technology, "Phoenix Events Veranstaltungstechnik"
Oct. 2005, 2006	Freelancer for "Suntours", promoting sailing boat trips
Sept. 2005, 2006, 2009	Innkeeper at the alpine hut "Schiestlhaus" on top of the mountain Hochschwab, Austria
Oct. 2001 - Sept. 2002	Paramedic at the Austrian Red Cross (civil service)
Sept. 2001	Working at the sculpture "Drama" by Austrian artist Franz West

since 1999	Freelancer as sound and light engineer at "Fantasy Veranstaltungstechnik GmbH" (since 2004: "Tent & Technic Veranstaltungsservice GmbH")
July 1998	Vacation job at archaeological excavations at the Mallnitzer Tauern, Austria

Additional Skills

German	mother tongue
English, Spanish	fluent written and spoken
French	fairly good written and spoken
IT skills	Good programming skills in Java, C++, Mosel, Pascal
	Cisco Certified Network Associate, Cisco Certified Network Professional, both successfully completed at school level at the TGM
	Solid knowledge in application of Microsoft Office (Word, Excel +Solver, Powerpoint), LaTeX
since 1989	Member of the boy scout group 16 "Schotten" Vienna, since 2000 as a voluntary leader
1998 - 2000	Voluntary leader at the Catholic youth centre "Schotten", Vienna
Driving Licenses	A,B,C,E,F

March 2012